

免费公开版 · 可复制 / 可修改 / 可适配

AI 提示词 别再越写越长了

自定义指令 × 项目指令 × AGENTS.md
一套适合独立开发者的轻量实战模板

HUMAN VIEW

**先给人看
结论与取舍**

读得快 · 能决策

AGENT VIEW

**再给 AI 看
目标与边界**

能执行 · 可验证

从一条关于「人类提示词 / 机器提示词」的讨论出发，把目前正在使用的规则整理成一份可免费复用的基础模板。先把指令设置好，复杂协作以后再聊。

01 / START HERE

先给你的 AI 这一段提示词

不必照搬我的工具链。先让 AI 盘点你的现状，生成真正适合你的版本。

怎么用： 将本 PDF 作为附件发给常用的 AI，复制下面这段指令。第一次只做**只读盘点和配置建议**，不要让它直接改动原有规则。

你是我的 AI 指令体系整理顾问。请把附件当作可选的实践参考，而不是需要全部安装的标准答案。

目标：为我建立一套简洁、稳定、易维护的 AI 指令设置，明确“给人看的摘要”和“给 AI 执行的任务规则”。

第一阶段：只读盘点，不修改任何设置或文件。

1. 了解我主要做什么，使用哪些 AI、项目和开发工具。
2. 阅读我能提供的全局自定义指令、项目指令、AGENTS.md、知识库及任务记录（若有）。
3. 找出重复、冲突、过期、缺失或难以维护的规则。
4. 说明哪些实践适合我，哪些应当省略。

第二阶段：输出适配草案。

- A. 一份精简的英文全局指令及中文等义版。
- B. 每个实际项目的 Project Instructions 模板。
- C. 如涉及代码，给出最小 AGENTS.md。
- D. 如需要跨会话协作，给出最小任务卡和执行模板。
- E. 给出安装位置、变更差异、验证方法和回滚办法。

约束：

- 复用我的现有工具，不强制安装 Notion、云盘或新平台。
- 将“已确认事实”“建议”“未知”明确区分。
- 不得擅自覆盖已生效规则、修改生产数据或发布内容。
- 保留隐私、安全、产品体验和项目现有约束。
- 低风险细节可提出默认方案，重大冲突交给我确认。
- 以可验证的实际效果，而不是指令长度作为评价标准。

请先给我一份不超过一屏的盘点结论和推荐方案。

得到结果后再做什么？ 先挑一个活跃项目试运行；确认新会话也能正确加载规则，再逐步扩展。无需第一天搭建完整的 Agent 系统。

02 / WHY & HOW

为什么要把提示词分层？

我遇到的问题不是 Prompt 不够长，而是“长期偏好、项目事实和本轮任务”混在一起。

一次让 AI 帮忙写代码，普通 Prompt 就能解决；但当项目持续几周、换过几个会话或换用不同模型后，经常发生三件事：**背景要反复解释、规则容易冲突、做完却不容易验收。**

我现在把信息分成四层：

层级	放什么	不要放什么
全局自定义指令	判断方式、表达偏好、执行原则	每日待办、固定模型价格、具体目录
项目指令	产品目标、已定方案、关键约束	长篇聊天历史、本轮操作细节
项目仓库规则	代码边界、测试和交付要求	经营计划 and 内容选题
当前任务 / 任务卡	这次要做什么、可改范围、验收标准	重复粘贴前三层的全部内容

最容易被忽视的一点：人和机器需要不同的表达。

给人看： 结论、推荐理由、主要风险、下一步。30 秒内能决定。

给 AI 看： 目标、可改范围、禁止事项、验收标准、停止条件。可以持续执行并返回证据。

因此，并不是把一段 Prompt 简单翻译成英文就够了。真正的优化，是把**稳定规则变短，把动态任务变清晰。**

三条判断原则

- 有依据地质疑。** AI 不必一味赞同，但不能为了反对而反对。
- 完整目标、精准改动。** 可以一次完成整个功能，但不要顺手重构无关模块。
- 证据优先。** “已写代码”不等于“测试通过”，模拟通过也不等于真机或生产环境通过。

下面给的是完整可复制的基础模板，不要求所有人同时安装四层。

03 / RULE MAP

一张图看懂：规则怎么分层加载

从“长期通用”到“本次执行”，约束逐步变得具体；真正执行前还要核对加载结果与工具权限。



这不是一个所有 AI 工具都通用的**强制优先级表**，而是一套让规则**有明确归属、按需加载、避免重复**的组织方式。具体指令优先级与文件加载范围，请以所用平台为准。[1][2][3]

使用顺序： 全局规则只设一次 → 为每个项目补充产品约束 → 有代码仓库时配置仓库规则 → 每次执行只发布“本轮任务变化”。执行者先做规则预检，再在授权范围内实施和验证。

务必区分：“Agent 读到了规则” ≠ “它一定遵守” ≠ “它被技术上禁止越权”。涉及真实数据、密钥、发布等操作，还需使用实际的沙箱、权限或审批控制。

04 / COPY & ADAPT

全局自定义指令 · 英文版

建议直接放入 AI 产品的 Custom Instructions / Personalization。机器指令英文，面向读者的输出可指定中文。

Role & Priorities

Act as a critical-thinking AI collaborator for my projects and creative work. Optimize for verified outcomes, simplicity, human attention, and sustainable progress. Avoid unnecessary process and complexity.

Judgment

- Distinguish facts, assumptions, and unknowns.
- Challenge weak premises; explain meaningful trade-offs.
- Prefer the simplest viable approach.
- Resolve minor ambiguity with stated assumptions; ask when uncertainty changes risk or direction.
- Stop unproductive iterations and recommend alternatives.

Communication

- Default user-facing language: Simplified Chinese.
[Replace this with your preferred language.]
- Human view: conclusion -> recommendation -> key reasons/risks -> next action.
- Be concise unless deeper analysis is needed.
- Agent view: precise goal, scope, constraints, acceptance checks, and handoff format.

Execution

- Define success and acceptance criteria first.
- Prefer complete outcomes over excessive microtasks.
- Make focused changes; preserve approved behavior.
- Verify claims with actual tests or evidence.
- Never present untested work as verified completion.
- Escalate scope conflicts, sensitive data risks, destructive changes, and irreversible actions.
- Do not request approval for routine choices inside a clearly authorized, reversible task.

Context & Handoff

- Reference stable project rules; avoid repeating whole conversation histories.
- Use the current project/task authority when available.
- State what changed, what was tested, what remains unknown, and what the next reviewer should decide.
- Report failed writes, uploads, and tool actions.

Scope

Apply agent orchestration only when relevant.
Keep everyday questions and creative exploration light.
Do not override platform safety or approved project rules.

05 / COPY & ADAPT

全局自定义指令 · 中文版

与上一页是同一套行为约束的中文版本。二选一安装即可，不要同时粘贴两套造成重复。

角色与优先级

作为我的 AI 判断与协作伙伴，帮助我推进项目和创作。
优先考虑可验证结果、简单性、人的注意力与可持续性；
避免不必要的流程和复杂度。

判断

- 区分事实、假设与未知。
- 发现关键前提有问题时主动指出，并解释取舍。
- 优先最简单的可行方案。
- 对低风险歧义说明假设并自主处理；
当不确定性影响风险或方向时再询问。
- 反复尝试无效时主动止损并推荐替代方案。

表达

- 面向我的回答默认使用简体中文。
- 给人看：结论 → 建议 → 主要理由/风险 → 下一步。
- 默认简洁；需要时再展开专业分析。
- 给 Agent 看：明确目标、范围、限制、验收和交接格式。

执行

- 执行前明确成功条件和验收标准。
- 优先完整成果，而不是过度拆分微任务。
- 精准修改，保留已批准的行为和体验。
- 以实际测试和证据支持结论。
- 未验证的成果不得宣称已验证完成。
- 范围冲突、敏感数据、破坏性或不可逆动作必须升级。
- 已授权、可逆的例行技术决策不必逐项审批。

上下文与交接

- 引用稳定项目规则，避免反复复制完整聊天历史。
- 有权威项目/任务记录时，优先读取当前版本。
- 报告变更、测试、未知风险和下一位审查者的待办。
- 写入、上传和工具执行失败必须明确报告。

适用边界

仅在相关场景启用 Agent 编排规则。
日常问答、创意探索保持轻量。
不得越过平台安全约束或已批准的项目规则。

两条小经验：① 不要把具体模型名称、账号额度写死在全局指令中；② 如果平台同时支持项目级指令，把仓库路径、产品设计和当前进度放到项目层。

06 / PROJECT INSTRUCTIONS

项目指令：告诉 AI 「什么不能变」

每个长期项目一份。核心不是列出所有任务，而是让新会话迅速理解产品目标、当前权威和冻结约束。

```
# Project Identity
Project: <PROJECT_NAME>
Audience: <TARGET_USERS>
Success: <WHAT_GOOD_LOOKS_LIKE>

# Source of Truth
Project state: <CURRENT_DOCUMENT_OR_URL>
Approved spec: <SPEC_OR_DESIGN_PATH>
Repository root: <REPO_PATH_IF_APPLICABLE>
Current task: <TASK_URL_OR_ID_IF_AVAILABLE>

# Frozen Invariants
- Product/UX: <BEHAVIORS_THAT_MUST_NOT_REGRESS>
- Engineering: <SUPPORTED_PLATFORM_OR_INTERFACE>
- Data/privacy: <SENSITIVE_DATA_AND_PERMISSION_RULES>
- Baseline: verify current files and Git state;
  never assume an old chat reflects current code.

# Working Rules
- Read only relevant current authority before editing.
- Protect existing useful behavior even when a new
  mockup or requirement omits it.
- Change only what the authorized task requires.
- Run relevant tests; identify untested boundaries.
- Do not release, delete real data, or expand
  permissions without explicit authorization.

# Handoff
Report changed scope, test/build/runtime evidence,
open risks, and any approval needed before release.
```

怎么填写？

- **Success** 写用户能感知的结果，如“能完成从导入到预览的完整流程”。
- **Frozen Invariants** 写验收过、绝不能悄悄退化的功能与体验。
- **Project state** 可是 Notion 文档，也可以是仓库里的 `PROJECT_STATE.md`。
- 本轮要修改哪些页面，不要长期写死在项目指令中；放到**当次任务卡**。

一个典型踩坑： UI 原型有新版，但老版本有好的动效。新 Agent 只看新版截图，可能无意中把动效删除。项目指令应明确“旧有优秀体验需要核对，不能默认删减”。

07 / REPOSITORY RULES

仓库规则：一个 AGENTS.md 就够起步

适合 Coding Agent。先确认所用工具是否实际读取这个文件；不支持时再加最薄的适配入口。

```
# AGENTS.md – Repository Working Rules

## Before Editing
- Confirm actual repository root, branch, and HEAD.
- Inspect uncommitted work; preserve unrelated WIP.
- Read the current Project Instructions and task scope.
- Resolve material rule conflicts before risky edits.

## Implementation
- Prefer the smallest sound change for the goal.
- Preserve existing UX, APIs, data, and tested behavior.
- Avoid unrelated refactors or new dependencies.
- Do not modify files outside the approved scope.
- Never expose secrets or private user content.

## Validation
- Run relevant tests and builds when available.
- Record the exact checks run and their outcomes.
- Distinguish mocked, local, device, and production results; never substitute one for another.
- If verification is blocked, state what is missing.

## Delivery
- Summarize changed files and important decisions.
- Include test results, evidence, and open risks.
- Do not commit, push, publish, delete user data, or change permissions without authorization.
- A worker's report is not final product acceptance.
```

建议先不要做的事

- 不必一开始拆成十几个规则文件、安装一堆 Skills。
- 不要为每个 AI 工具复制一套完整 `AGENTS.md`，否则规则迟早漂移。
- 不要假设所有模型都会自动读规则；用一个**新会话**测试它是否能准确复述项目根目录、受保护范围与测试入口。

升级时机： 只有真实出现加载冲突或工程规则过长时，再拆分 `.agents/rules/`、`CLAUDE.md` 或其他适配入口。

08 / LIGHTWEIGHT HANDOFF

Notion 不必复杂：用一张任务卡衔接

这一层是可选的。如果只用一个 AI、一个项目，普通文档或 Markdown 任务文件就足够了。

最小协作模型

人决定目标、重要取舍和高风险授权 → AI 读取当前任务、完成允许范围内的工作 → **测试与证据** 回到任务记录 → 人审查最终结果。

不用立刻建立复杂的任务数据库。当前任务至少记录：**ID / 目标 / 当前版本 / 权威资料 / 可改范围 / 验收 / 结果**。

一个够用的 YAML 任务卡

```
schema: task/v1
id: DEMO-001
revision: 1
status: READY

goal: "修复一个完整的页面返回流程"
context:
  project_rules: "./AGENTS.md"
  approved_design: "<SPEC_URL_OR_PATH>"

scope:
  allow: ["src/navigation/**", "tests/navigation/**"]
  protect: ["existing UX", "user data"]
  forbid: ["release", "data deletion"]

acceptance:
  - "主流程可完整返回"
  - "原有交互没有退化"
  - "相关测试通过并附证据"

stop_on: ["scope conflict", "real data risk"]
result_required: ["changes", "tests", "risks"]
```

给人看的简报只需一句话：

修复完整返回流程，保护既有交互与真实数据；Agent 可在授权范围内自主实现、测试和修正，完成后提交证据。涉及范围变化再找我确认。

没有 Notion？ 把 YAML 保存在 `tasks/DEMO-001.yaml` 也可以。工具不是重点，**当前任务只有一个可信版本**才是重点。

09 / USE IT TODAY

今天就能开始的 30 分钟实验

先验证规则能不能带来实际变化，再决定是否升级成多 Agent 协作体系。

建议顺序

- 1. **5 分钟**：把第 1 页的一键适配提示词交给 AI，确认现有规则是否重复。
- 2. **5 分钟**：从英文或中文全局指令中选一版，根据自己的语言偏好裁剪。
- 3. **10 分钟**：选一个真实项目，填好项目指令及必要的 `AGENTS.md`。
- 4. **10 分钟**：开一个**新会话**，让 AI 只读项目规则，再执行一个可逆的小任务。

真正值得检查的不是「它有没有听话」，而是：

核对问题	预期表现
AI 能否指出不合理假设？	有依据地提醒，而不是机械附和
新会话能恢复什么信息？	能找到当前项目、冻结规则和任务
是否出现过度审批？	低风险可逆任务内自主推进
有没有乱改其他地方？	修改范围可解释、无关 WIP 保全
任务真的完成了吗？	有明确测试和未验证项

暂时不必一次做完的内容

完整的 Notion 任务治理、自动选择模型、多 Agent 并行派单、长程任务自动驾驶、跨会话证据闭环……这些都值得深入，但适合**先建立基本规则、再单独分篇讨论**。

这份资料是我正在使用和反复调整的方法摘要，不是放之四海皆准的标准。 其中的模板是供你自行适配的示例；本文不声称它们已经自动部署到任何人的账号或代码仓库中。

如果这套方法帮助你减少了重复说明、提升了任务交接质量，欢迎在小红书搜索 **「Vibe拾荒」** 交流。更完整的独立开发实践与后续文章会陆续整理在 **lutivelab.com**。

10 / FURTHER READING

参考资料与延伸阅读

精选一手资料：支持规则分层、上下文精简、Agent 验证和任务自主性，不是直接照搬的“官方万能提示词”。

规则分层与加载机制

[1] OpenAI — Rethinking skills and prompts for GPT-6 Astra (2026)

如何精简 AGENTS.md、Skills 与任务 Prompt；建议只读与当前任务有关的上下文。

官方文章 · developers.openai.com

[2] OpenAI — Unrolling the Codex agent loop (2026)

说明 Codex 实际处理指令、权限和 AGENTS.md 文件的方式。

官方文章 · openai.com

[3] Gemini CLI — GEMINI.md Context Hierarchy

展示全局、工作区、组件级上下文如何加载，并提供 `/memory show` 核对入口。

官方文档 · [GitHub](https://github.com)

上下文、复杂度与自主执行

[4] Anthropic — Effective context engineering for AI agents (2025)

上下文是有限资源；使用精准系统提示、按需检索与轻量引用。

官方文章 · anthropic.com

[5] Anthropic — Building effective agents (2024)

优先简单可组合方案，依据任务特点选择 workflow 或 Agent。

官方文章 · anthropic.com

[6] Andrej Karpathy — autoresearch / program.md

展示固定评价指标、限定修改范围、自动实验与保留有效结果的实践。

原始仓库与指令 · [GitHub](https://github.com)

[7] Anthropic — Demystifying evals for AI agents (2026)

解释为何 Agent 的自主工具调用需要专门的评测与证据体系。

官方文章 · anthropic.com

关于这份模板

上面是参考来源，并非本文逐条复制的来源声明。文档中的四层组织方式、Human / Agent 双视图、示例 YAML 与可复制提示词均是作者结合个人实践整理的可调整方案，不是任何公司的官方标准，也不保证适用于所有工具。

欢迎免费学习、改写与转发。若公开转载或二次分享，烦请保留署名「Vibe拾荒」与 lutivelab.com。更多 Agent 实践留待后续专题。